On the Robustness and Scalability of SemidefiniteRelaxation for Optimal Power Flow Problems

Eltved, Anders; Dahl, Joachim ; Andersen, Martin Skovgaard

Published in:
Optimization and Engineering

Link to article, DOI:
10.1007/s11081-019-09427-4

Publication date:
2020

Document Version
Peer reviewed version

Noname manuscript No.
(will be inserted by the editor)

On the Robustness and Scalability of Semidefinite Relaxation for Optimal Power Flow Problems

Anders Eltved · Joachim Dahl · Martin S. Andersen

the date of receipt and acceptance should be inserted later

Abstract Semidefinite relaxation techniques have shown great promise for nonconvex optimal power flow problems. However, a number of independent numerical experiments have led to concerns about scalability and robustness of existing SDP solvers. To address these concerns, we investigate some numerical aspects of the problem and compare different state-of-the-art solvers. Our results demonstrate that semidefinite relaxations of large problem instances with on the order of 10,000 buses can be solved reliably and to reasonable accuracy within minutes. Furthermore, the semidefinite relaxation of a test case with 25,000 buses can be solved reliably within half an hour; the largest test case with 82,000 buses is solved within eight hours. We also compare the lower bound obtained via semidefinite relaxation to locally optimal solutions obtained with nonlinear optimization methods and calculate the optimality gap.

Keywords AC Optimal Power Flow · Semidefinite Relaxation · Optimization · Numerical Analysis

1 Introduction

The alternating current optimal power flow (ACOPF) problem is a nonlinear optimization problem that is concerned with finding an optimal operating point for a power system network. Today, more than 60 years after it was first studied by Carpentier (1962), the problem still receives considerable attention because of the challenging nature of the problem and its important

A. Eltved and M. S. Andersen
Department of Applied Mathematics and Computer Science, Technical University of Denmark, 2800 Kgs. Lyngby, Denmark (e-mail: {aelt,mskan}@dtu.dk)

J. Dahl
MOSEK ApS, Fruebjergvej 3, Symbion Science Park, 2100 Copenhagen, Denmark (e-mail: dahl.joachim@gmail.com)

role in power system planning and operation. Many optimization methods have been applied to the ACOPF problem, including general nonlinear optimization techniques, interior-point methods, and meta-heuristic optimization methods (Taylor, 2015).

Following the work of Jabr (2006) and Bai et al. (2008), the use of convex relaxation techniques applied to the ACOPF problem has been explored extensively; see *e.g.* (Low, 2014a,b) for a recent survey. The interest in these techniques is driven by the fact that the solution to a relaxed problem provides either a globally optimal solution to the original problem or a global lower bound that can be used to assess the quality of locally optimal solutions found by other means. Moreover, a solution to an SDR may also be used to guide a load flow study (Mak et al., 2018) in order to find a feasible operating point.

Different convex relaxations of the ACOPF problem have been proposed and studied, including a second-order cone relaxation (SOCR) (Jabr, 2006), a semidefinite relaxation (SDR) (Bai et al., 2008; Lavaei and Low, 2012), moment relaxations (Molzahn and Hiskens, 2015; Josz et al., 2015), and more recently, a quadratic convex relaxation (QCR) (Coffrin et al., 2016; Hijazi et al., 2017). The different relaxations vary in tightness and computational cost; we refer to (Coffrin et al., 2016) for a recent comparison of the SDR, QCR, and SOCR. For example, the SDR is generally tighter than the SOCR, but it is generally also more computationally demanding. One direction of research is dedicated to strengthening the SOCR; see *e.g.* (Kocuk et al., 2016). In an attempt to address the computational cost associated with the SDR, Andersen et al. (2014) and Bingane et al. (2018) have proposed simpler, weaker SDRs that are cheaper to solve than the standard SDR. The QCR is generally neither weaker nor stronger than the SDR, but it is computationally cheaper and often provides a lower bound of similar quality as that of the SDR.

The high computational cost of solving an SDR of a large ACOPF problem has given rise to concerns about robustness and scalability (Hijazi et al., 2016, 2017; Madani et al., 2017). These concerns are supported by numerical experiments that show that solving the SDR is not only much slower than other approaches, but also more unreliable (Coffrin et al., 2016). Our goal with this paper is to address concerns regarding robustness and scalability by demonstrating numerically that an SDR of the ACOPF problem can be solved both reliably and within minutes using commodity hardware, even for large networks with on the order of 10,000 buses. Our contribution is therefore confined to numerical considerations and implementation details (Section 2) as well as numerical experiments (Section 3) with the purpose of investigating scalability, accuracy, and robustness for different solvers. What differentiates our implementation from most implementations that have been described and investigated in the literature is the fact that we construct the SDR manually without the use of modeling tools such as YALMIP (Löfberg, 2004) and CVX (Grant and Boyd, 2008). Although this manual approach can be both inflexible and cumbersome, it is typically much faster and allows us to control the exact problem formulation, avoiding automatic transformations that may ad-

versely affect the size and conditioning of the SDR problem. We remark that some modeling tools allow some degree of control over the problem formulation (*e.g.*, through options), but it is generally difficult for non-expert users to predict the final problem formulation.

Notation The set $\mathcal{K}_{\mathrm{q}}^n = \{(t,x) \in \mathbb{R} \times \mathbb{R}^{n-1} \,|\, \|x\|_2 \leq t\}$ denotes the second-order cone in $\mathbb{R}^n$, $\mathbb{S}^n$ denotes the set of symmetric matrices of order n, and $\mathbb{H}^n$ is the set of Hermitian matrices of order n. The sets $\mathbb{S}_+^n$ and $\mathbb{H}_+^n$ are the cones of positive semidefinite matrices in $\mathbb{S}^n$ and $\mathbb{H}^n$, respectively. Since the symmetric matrices of order n form a vector space of dimension $n(n+1)/2$, the cone $\mathbb{S}_+^n$ can be reparameterized as $\mathcal{K}_{\mathrm{s}}^n = \{\mathbf{svec}(X) \,|\, X \in \mathbb{S}_+^n\} \subset \mathbb{R}^{n(n+1)/2}$ where $\mathbf{svec}(\cdot)$ is an injective function that maps a symmetric matrix of order n to a vector of length $n(n+1)/2$. Similarly, we define $\mathcal{K}_{\mathrm{h}}^n = \{\mathbf{hvec}(X) \,|\, X \in \mathbb{H}_+^n\} \subset \mathbb{R}^{n^2}$ where $\mathbf{hvec}(\cdot)$ maps a Hermitian matrix of order n to a vector of length n^2. The inner product between two matrices $A, B \in \mathbb{H}^n$ is $\mathbf{tr}(A^H B)$ where $\mathbf{tr}(A)$ denotes the trace of a square matrix A. Given a complex number $c = a + \jmath b$ where $\jmath = \sqrt{-1}$, $\Re(c)$ denotes the real part a, $\Im(c)$ denotes the imaginary part b, and c^* denotes the complex conjugate of c.

2 Method

2.1 The AC Optimal Power Flow Problem

An AC power system in steady state can be modeled as a directed graph where the set of nodes $\mathcal{N} = \{1, 2, \ldots, n\}$ corresponds to a set of n power buses, and the set of edges $\mathcal{L} \in \mathcal{N} \times \mathcal{N}$ corresponds to transmission lines, *i.e.*, $(k,l) \in \mathcal{L}$ if there is a line from bus k to bus l. The set $\mathcal{L}^{\mathrm{fl}} \subseteq \mathcal{L}$ consists of all transmission lines with a flow constraint, $\mathcal{L}^{\mathrm{pa}} \subseteq \mathcal{L}$ consists of all transmission lines with a phase-angle difference constraint, $\mathcal{G}_k$ denotes a (possibly empty) set of generators associated with bus k, and $\mathcal{G} = \bigcup_{k \in \mathcal{N}} \mathcal{G}_k$ is the set of all generators. The power produced by generator $g \in \mathcal{G}$ is $s_g = p_g + \jmath q_g$, and at each power bus $k \in \mathcal{N}$, we define a complex load (*i.e.*, demand) $S_k^d = P_k^d + \jmath Q_k^d$, a complex voltage v_k, and a complex current i_k. To simplify notation, we define a vector of voltages $v = (v_1, v_2, \ldots, v_n)$ and a vector of currents $i = (i_1, i_2, \ldots, i_n)$. With this notation, the ACOPF problem can be expressed as

$$\text{minimize} \quad \sum_{g \in \mathcal{G}} f_g(p_g) \tag{1a}$$

subject to

$$i_k^* v_k = \sum_{g \in \mathcal{G}_k} s_g - S_k^d, \qquad k \in \mathcal{N} \tag{1b}$$

$$P_g^{\min} \leq p_g \leq P_g^{\max}, \qquad g \in \mathcal{G} \tag{1c}$$

$$Q_g^{\min} \leq q_g \leq Q_g^{\max}, \qquad g \in \mathcal{G} \tag{1d}$$
$$V_k^{\min} \leq |v_k| \leq V_k^{\max}, \qquad k \in \mathcal{N} \tag{1e}$$
$$\left|S_{k,l}^{\mathrm{fl}}(v)\right| \leq S_{k,l}^{\max}, \qquad (k,l) \in \mathcal{L}^{\mathrm{fl}} \tag{1f}$$
$$\left|S_{l,k}^{\mathrm{fl}}(v)\right| \leq S_{l,k}^{\max}, \qquad (k,l) \in \mathcal{L}^{\mathrm{fl}} \tag{1g}$$
$$\phi_{k,l}^{\min} \leq \angle(v_k v_l^*) \leq \phi_{k,l}^{\max}, \qquad (k,l) \in \mathcal{L}^{\mathrm{pa}} \tag{1h}$$
$$i = Yv \tag{1i}$$

with variables $i \in \mathbb{C}^n$, $v \in \mathbb{C}^n$, and $s \in \mathbb{C}^{|\mathcal{G}|}$, and where $i = Yv$ corresponds to Ohm's law in matrix form, given the network admittance matrix $Y \in \mathbb{C}^{n \times n}$. The cost of generation for generator g is given by $f_g(p_g)$, and we will restrict our attention to convex quadratic generation cost functions, *i.e.*,

$$f_g(p_g) = \alpha_g p_g^2 + \beta_g p_g + \gamma_g, \tag{2}$$

where the parameters $\alpha_g \geq 0$, β_g, and γ_g are given. The constraints (1b) are power balance equations, (1c) and (1d) are generation limits, (1e) are voltage magnitude limits, (1f) and (1g) are transmission line flow constraints, and (1h) are phase-angle difference constraints. The flow from bus k to bus l is given by $S_{k,l}^{\mathrm{fl}}(v) = v^H T_{k,l} v + \jmath v^H \widetilde{T}_{k,l} v$ (provided that $(k,l) \in \mathcal{L}^{\mathrm{fl}}$ or $(l,k) \in \mathcal{L}^{\mathrm{fl}}$) where $T_{k,l} \in \mathbb{H}^n$ and $\widetilde{T}_{k,l} \in \mathbb{H}^n$ are given.

2.2 Semidefinite Relaxation

Roughly following the steps described in (Andersen et al., 2014), we start by reformulating the ACOPF problem (1). Specifically, we perform the following steps:

1. Eliminate $i = Yv$ and substitute $P_g^{\min} + p_g^{\mathrm{l}}$ for p_g, $Q_g^{\min} + q_g^{\mathrm{l}}$ for q_g, and X for vv^H.
2. Drop constant terms in the objective:

$$\begin{aligned} f(p_g) &= \alpha_g (P_g^{\min} + p_g^{\mathrm{l}})^2 + \beta_g (P_g^{\min} + p_g^{\mathrm{l}}) + \gamma_g \\ &= \alpha_g (p_g^{\mathrm{l}})^2 + \tilde{\beta}_g p_g^{\mathrm{l}} + \text{const.} \end{aligned}$$

   where $\tilde{\beta}_g = (\beta_g + 2\alpha_g P_g^{\min})$.
3. Introduce an auxiliary variable t_g for each $g \in \mathcal{G}^{\mathrm{quad}} = \{g \in \mathcal{G} \,|\, \alpha_g > 0\}$ and include epigraph constraint

$$\alpha_g (p_g^{\mathrm{l}})^2 \leq t_g \;\Leftrightarrow\; \begin{bmatrix} 1/2 + t_g \\ 1/2 - t_g \\ \sqrt{2\alpha_g}\, p_g^{\mathrm{l}} \end{bmatrix} \in \mathcal{K}_{\mathrm{q}}^3.$$

4. Introduce slack variables to obtain a standard-form formulation.

These steps yield the equivalent problem

$$\text{minimize} \quad \sum_{g\in\mathcal{G}} \tilde{\beta}_g p_g^{\mathrm{l}} + \sum_{g\in\mathcal{G}^{\mathrm{quad}}} t_g \tag{3a}$$

subject to

$$\begin{aligned}
\mathbf{tr}(Y_k X) &= \sum_{g\in\mathcal{G}_k} (P_g^{\min} + p_g^{\mathrm{l}}) - P_k^d, && k \in \mathcal{N} && \text{(3b)}\\
\mathbf{tr}(\widetilde{Y}_k X) &= \sum_{g\in\mathcal{G}_k} (Q_g^{\min} + q_g^{\mathrm{l}}) - Q_k^d, && k \in \mathcal{N} && \text{(3c)}\\
p_g^{\mathrm{l}} + p_g^{\mathrm{u}} &= P_g^{\max} - P_g^{\min}, && g \in \mathcal{G} && \text{(3d)}\\
q_g^{\mathrm{l}} + q_g^{\mathrm{u}} &= Q_g^{\max} - Q_g^{\min}, && g \in \mathcal{G} && \text{(3e)}\\
X_{kk} - \nu_k^{\mathrm{l}} &= (V_k^{\min})^2, && k \in \mathcal{N} && \text{(3f)}\\
X_{kk} + \nu_k^{\mathrm{u}} &= (V_k^{\max})^2, && k \in \mathcal{N} && \text{(3g)}\\
z_{k,l} &= \begin{bmatrix} S_{k,l}^{\max} \\ \mathbf{tr}(T_{k,l} X) \\ \mathbf{tr}(\widetilde{T}_{k,l} X) \end{bmatrix}, && (k,l) \in \mathcal{L}^{\mathrm{fl}} && \text{(3h)}\\
z_{l,k} &= \begin{bmatrix} S_{l,k}^{\max} \\ \mathbf{tr}(T_{l,k} X) \\ \mathbf{tr}(\widetilde{T}_{l,k} X) \end{bmatrix}, && (k,l) \in \mathcal{L}^{\mathrm{fl}} && \text{(3i)}\\
w_g &= \begin{bmatrix} 1/2 + t_g \\ 1/2 - t_g \\ \sqrt{2\alpha_g} p_g^{\mathrm{l}} \end{bmatrix}, && g \in \mathcal{G}^{\mathrm{quad}} && \text{(3j)}\\
\Im(X_{kl}) &= \tan(\phi_{k,l}^{\min}) \Re(X_{kl}) + y_{k,l}^{\mathrm{l}}, && (k,l) \in \mathcal{L}^{\mathrm{pa}} && \text{(3k)}\\
\Im(X_{kl}) &= \tan(\phi_{k,l}^{\max}) \Re(X_{kl}) - y_{k,l}^{\mathrm{u}}, && (k,l) \in \mathcal{L}^{\mathrm{pa}} && \text{(3l)}\\
p_g^{\mathrm{l}}, p_g^{\mathrm{u}} &\geq 0, && g \in \mathcal{G} && \text{(3m)}\\
q_g^{\mathrm{l}}, q_g^{\mathrm{u}} &\geq 0, && g \in \mathcal{G} && \text{(3n)}\\
\nu_k^{\mathrm{l}}, \nu_k^{\mathrm{u}} &\geq 0, && k \in \mathcal{N} && \text{(3o)}\\
y_{k,l}^{\mathrm{l}}, y_{k,l}^{\mathrm{u}} &\geq 0, && (k,l) \in \mathcal{L}^{\mathrm{pa}} && \text{(3p)}\\
z_{k,l}, z_{l,k} &\in \mathcal{K}_{\mathrm{q}}^3, && (k,l) \in \mathcal{L}^{\mathrm{fl}} && \text{(3q)}\\
w_g &\in \mathcal{K}_{\mathrm{q}}^3, && g \in \mathcal{G}^{\mathrm{quad}} && \text{(3r)}\\
X &= vv^H && && \text{(3s)}
\end{aligned}$$

with variables $p^{\mathrm{l}}, p^{\mathrm{u}}, q^{\mathrm{l}}, q^{\mathrm{u}} \in \mathbb{R}^{|\mathcal{G}|}$, $t \in \mathbb{R}^{|\mathcal{G}^{\mathrm{quad}}|}$, $\nu^{\mathrm{l}}, \nu^{\mathrm{u}} \in \mathbb{R}^{|\mathcal{N}|}$, $y^{\mathrm{l}}, y^{\mathrm{u}} \in \mathbb{R}^{|\mathcal{L}^{\mathrm{pa}}|}$, $z_{k,l}, z_{l,k} \in \mathcal{K}_{\mathrm{q}}^3$ for $(k,l) \in \mathcal{L}^{\mathrm{fl}}$, $w_g \in \mathcal{K}_{\mathrm{q}}^3$ for $g \in \mathcal{G}^{\mathrm{quad}}$, $X \in \mathbb{H}^n$, and $v \in \mathbb{C}^n$. Notice that the constraints (3b)-(3l) are all linear. We refer the reader to (Andersen et al., 2014) for a definition of the data matrices $Y_k, \widetilde{Y}_k, T_{k,l}$, and $\widetilde{T}_{k,l}$.

The only non-convex constraint in (3) is the rank-1 condition (3s). An SDR of (3) is readily obtained by replacing (3s) by the positive semidefiniteness constraint $X \succeq 0$. The resulting SDR is a so-called cone linear program (CLP) that can be expressed as

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & x \in \mathcal{K} \end{array} \tag{4}$$

where x is the vector of variables and the cone $\mathcal{K}$ is a Cartesian product of three types of cones, *i.e.*,

$$\mathcal{K} = \mathbb{R}^{n_l}_+ \times \underbrace{\mathcal{K}^3_{\mathrm{q}} \times \cdots \times \mathcal{K}^3_{\mathrm{q}}}_{n_q} \times \mathcal{K}^n_{\mathrm{h}}.$$

Thus, the number of variables is $N = n_l + 3n_q + n^2$ where $n_l = 4|\mathcal{G}| + |\mathcal{G}^{\mathrm{quad}}| + 2|\mathcal{N}| + 2|\mathcal{L}^{\mathrm{pa}}|$ and $n_q = 2|\mathcal{L}^{\mathrm{fl}}| + |\mathcal{G}^{\mathrm{quad}}|$, and the number of equality constraints is $M = 4|\mathcal{N}| + 2|\mathcal{G}| + 2|\mathcal{L}^{\mathrm{pa}}| + 3n_q$.

2.3 Conversion

The computational cost of solving (4) with a general-purpose interior-point method becomes prohibitively large when n is large: the cost of an interior-point iteration is at least $O(n^3)$. Fortunately, the problem (4) is generally very sparse in practice, and hence the conversion method of Fukuda et al. (2001) may be used to rewrite (4) as an equivalent CLP

$$\begin{array}{ll} \text{minimize} & \tilde{c}^T \tilde{x} \\ \text{subject to} & \tilde{A}\tilde{x} = b \\ & E\tilde{x} = 0 \\ & \tilde{x} \in \widetilde{\mathcal{K}} \end{array} \tag{5}$$

with

$$\widetilde{\mathcal{K}} = \mathbb{R}^{n_l}_+ \times \underbrace{\mathcal{K}^3_{\mathrm{q}} \times \cdots \times \mathcal{K}^3_{\mathrm{q}}}_{n_q} \times \mathcal{K}^{r_1}_{\mathrm{h}} \times \cdots \times \mathcal{K}^{r_m}_{\mathrm{h}}.$$

The conversion essentially decomposes the cone $\mathcal{K}^n_{\mathrm{h}}$ into a Cartesian product of a number of lower-dimensional cones $\mathcal{K}^{r_1}_{\mathrm{h}} \times \cdots \times \mathcal{K}^{r_m}_{\mathrm{h}}$ at the expense of a set of coupling constraints $E\tilde{x} = 0$. This reformulation of the problem can have a dramatic effect on the computational cost of solving the SDR of the ACOPF problem, and it effectively mitigates the $O(n^3)$ bottleneck that arises with the formulation (4). Moreover, the conversion technique often induces sparsity in the system of equations that define the search direction at each interior-point iteration, reducing the cost per iteration further if the solver can exploit this type of sparsity. The conversion technique was first applied to SDRs of the ACOPF problem by Jabr (2012).

2.4 Implementation

Before turning to our numerical experiments, we briefly outline our implementation (Andersen, 2018). The code is written in Python and performs the following steps:

1. Read case file and build the CLP (4).
2. Apply conversion method: convert (4) to (5).
3. Apply Hermitian-to-symmetric transformation: map $\mathcal{K}_{\mathrm{h}}^{r_i}$ to $\mathcal{K}_{\mathrm{s}}^{2r_i}$ for $i = 1, \ldots, m$.
4. Scale the problem data to improve conditioning.

As part of the first step, we allow some preprocessing of the data: (i) slack variables p_g (or q_g) for which $P_g^{\min} = P_g^{\max}$ (or $Q_g^{\min} = Q_g^{\max}$) may be eliminated, (ii) numerical proxies for infinity which are used to indicate the absence of limits (*e.g.*, on generation) may be truncated, and (iii) a minimum resistance of transmission lines may be enforced. The Hermitian-to-symmetric transformation is a well known trick that is only necessary because the solvers used in our experiments cannot directly handle cones of Hermitian positive semidefinite matrices; see *e.g.* (Boyd and Vandenberghe, 2004). The scaling of the problem data in step 4 is a row-scaling of the equality constraints $\tilde{A}\tilde{x} = b$ in the CLP in (5). We define a vector α with elements

$$\alpha_i = \max\{\max_j |\tilde{A}_{ij}|, |b_i|, 1\}$$

and use the equivalent, scaled constraints $\mathbf{diag}(\alpha)^{-1}\tilde{A}\tilde{x} = \mathbf{diag}(\alpha)^{-1}b$. Additionally, we scale the objective to become $\tilde{c}^T\tilde{x}/\max\{\|\tilde{c}\|_2, 1\}$. This yields an equivalent problem, and we found that for some solvers, this can reduce the computational time by roughly a factor of two; we briefly return to the topic of scaling in Section 4.

3 Results

3.1 Experiments

To investigate the robustness and scalability of our methodology, we conducted a series of numerical experiments based on a collection of test cases from MATPOWER (Zimmerman et al., 2011) (which includes a number of test cases from (Josz et al., 2016)) and Power Grid Lib (PGLib-OPF, 2018) with as many as $n = 70{,}000$ power buses; we have also included a synthetic case of the continental USA from the Electric Grid Test Case Repository (Birchfield et al., 2017) with $n = 82{,}000$ power buses. We excluded cases that are infeasible and cases with generator cost functions that are neither quadratic nor linear. For each test case, we set up a CLP formulation of the SDR and solved it using five different CLP solvers: MOSEK 8.1 (MOSEK, 2015), SeDuMi 1.3 (Sturm, 1999), SDPT3 4.0 (Toh et al., 1999), SCS 1.2.7 (O'Donoghue et al.,

2016), and CDCS 1.1 (Zheng et al., 2016). MOSEK, SeDuMi, and SDPT3 are interior-point methods whereas SCS and CDCS are first-order methods based on the alternating direction method of multipliers (ADMM).

To compare our methodology to an approach based on a modeling tool, we used SDPOPF (Molzahn et al., 2013) from "MATPOWER Extras" to set up and solve an SDR of each case. SDPOPF uses YALMIP (Löfberg, 2004) to set up the problem which is then solved numerically using one of several possible solvers: we used MOSEK in order to facilitate a fair comparison. Finally, to compare our approach to a nonlinear optimization approach, we used MATPOWER to set up and solve each case with three different interior-point methods for nonlinear optimization: MIPS (Wang et al., 2007) from MATPOWER 6.1, IPOPT 3.12.9 (Wächter and Biegler, 2006) with PARDISO 6.0 (Kourounis et al., 2018), and KNITRO 10.3.1 (Byrd et al., 2006). These are all called via MATPOWER using its default initialization—the default is sometimes referred to as "flat start" since all voltages are set to 1 p.u. and the active power generation is set to the midpoint of its bounds. When successful, these solvers return a locally optimal solution that provides an upper bound on the optimal value in contrast to the SDR that provides a lower bound.

3.2 Setup

Using the implementation described in section 2.4, we processed the problem data before setting up the SDRs. Specifically, we truncated generator bounds larger than 50 times the base MVA. We remark that SDPOPF enforces a minimum transmission line resistance of 10^{-4}; in the experiments, we do not enforce a minimum resistance in our SDR.

All experiments but those involving KNITRO were conducted on an HPC node with two Intel XeonE5-2650v4 processors (a total of 24 cores) and 240 GB memory. All experiments with KNITRO were conducted on different hardware (2.5 GHz Intel Core i5 CPU, 8 GB of memory) because of license restrictions. As a result, the KNITRO computation times that we report cannot be compared directly to those reported for the other solvers. All MATLAB-based solvers were used with MATLAB R2017b, and MOSEK was called through its Python interface in Python 3.6.3. Finally, we modified the default solver options as follows: for SeDuMi, we raised the maximum number of iterations from 150 to 250; for SCS and CDCS, we limited the number of iterations to 20,000; for CDCS, we disabled "chordalize" and used the "primal" solver since this allowed us to solve the most cases; for SCS, we used the direct solver; for SDPT3 we used a value of 400 for "smallblockdim" and changed the maximum number of iterations from 100 to 250.

3.3 Robustness

We start with an investigation of robustness. Table 1 contains a summary of return statuses for the different solvers for a total of 159 test cases. The column

Table 1 Summary of return statuses by solver.

Solver	Success	Max. iter.	Failure
MOSEK	159	0	0
SeDuMi	53	0	106
SDPT3	52	0	107
SDPOPF	128	0	31
CDCS	146	13	0
SCS	17	142	0
IPOPT	133	0	26
KNITRO	145	0	14
MIPS	116	0	43

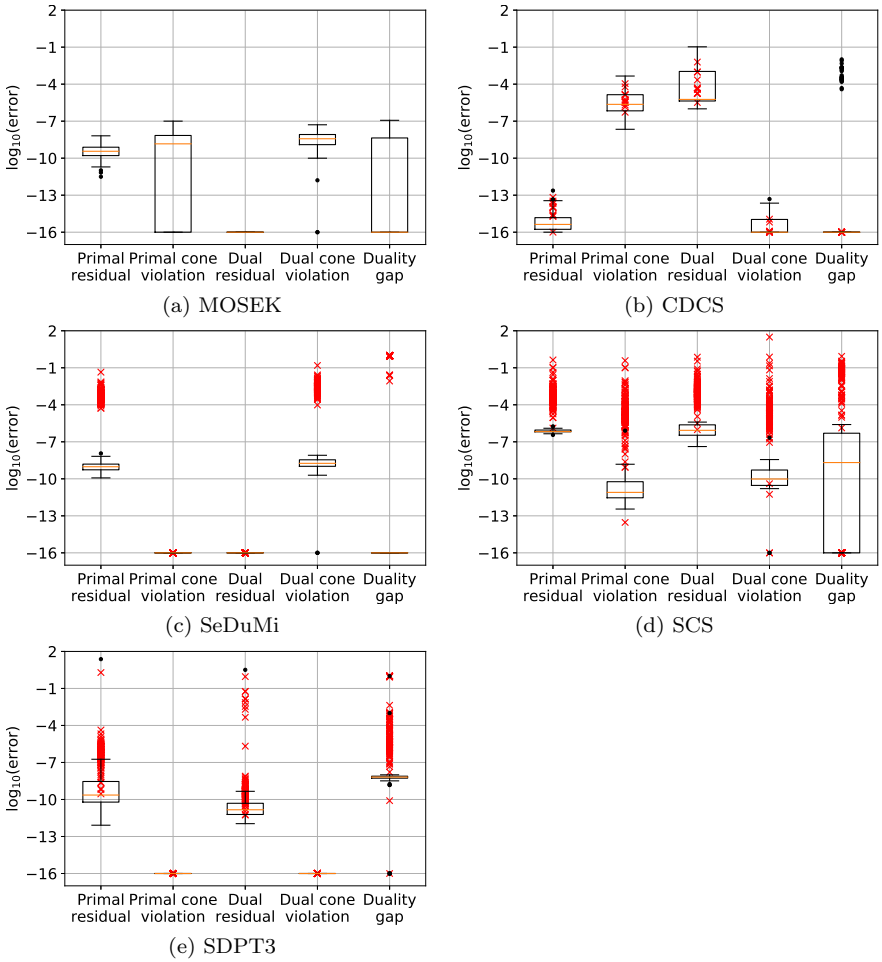
labeled "success" refers to return values that indicate successful termination with an optimal or near optimal (global or local) solution. The "failure" column refers to return values that indicate some kind of error. We remark that SDPOPF ignores phase angle constraints and fails in 31 cases because of a MATPOWER error; the solver is never called in these cases.

The results in Table 1 clearly demonstrate that the SDRs can be solved reliably using MOSEK: all cases were solved to optimality with MOSEK's default tolerances. In contrast, the nonlinear solvers IPOPT and KNITRO only succeed in roughly 85% of the cases while MIPS succeeds in approximately 75% of the cases. CDCS solves over 90% of the cases, but the accuracy and speed is poor compared to MOSEK as we show later in this section. Both SeDuMi and SCS succeed in less than 50% of the cases.

3.4 Accuracy

We now compare the solutions returned by the five CLP solvers. Since the solvers have different tolerances (*i.e.*, stopping criteria), we will compare the solvers based on the so-called "DIMACS error measures" described in (Mittelmann, 2003). Roughly speaking, these are five relative error measures quantifying the primal residual norm, primal cone violation, dual residual norm, dual cone violation, and duality gap. Fig. 1 summarizes the results in a box plot of the DIMACS measures for each solver (the smaller the error, the better).

MOSEK, shown in Fig. 1a, generally performs well with DIMACS errors below 10^{-7} in all cases. The SeDuMi errors, shown in Fig. 1c, reveal that SeDuMi returns a high-accuracy solution whenever it succeeds; the same is true for SDPT3, shown in Fig. 1e. This suggests that the default tolerances may be too strict for all but the small cases. Both CDCS and SCS generally return solutions with larger errors, as shown in Fig. 1b and 1d. This is to be expected since they are both first-order methods. While CDCS is relatively robust, it often terminates with sizable dual residuals which are indicative of low-accuracy solutions.



(a) MOSEK (b) CDCS

(c) SeDuMi (d) SCS

(e) SDPT3

Fig. 1 Box plots of logarithm of DIMACS errors. The red markers correspond to cases where the solver did not succeed. We note that in order to accommodate a logarithmic axis, we have replaced errors below 10^{-16} by this value.

3.5 Optimality Gap

Next we investigate the objective values provided by the solvers. We limit our attention to MOSEK and the nonlinear solvers IPOPT, MIPS, and KNITRO. The nonlinear solvers provide an upper bound when they terminate at a feasible point. We define the best upper bound as

$$\overline{f} = \min(f_{\text{IPOPT}}, f_{\text{KNITRO}}, f_{\text{MIPS}}), \tag{6}$$

i.e., the minimum of the objective values provided by the three solvers (if a solver does not succeed, we define its objective value to be ∞). Similarly, the

SDR (MOSEK) provides a lower bound which we denote by $\underline{f} = f_{\text{MOSEK}}$. The optimality gap may then be defined as

$$\text{gap} = \frac{\overline{f} - \underline{f}}{\overline{f}} \cdot 100\%. \tag{7}$$

The gap is equal to 0 if $\underline{f} = \overline{f}$, implying that we have a globally optimal solution. On the other hand, if the gap is large, $\overline{f}$ may be a poor local minimum and/or the SDR provides a weak lower bound $\underline{f}$.

We have made four tables listing objective values and optimality gap for all cases with more than 300 buses based on their origin: table 2 contains cases from the MATPOWER library; table 3 contains cases from PGLIB in typical operating conditions; table 4 contains cases from PGLIB with small phase angle difference constraints; table 5 contains cases from PGLIB with binding thermal limit constraints. The cases are sorted by the number of buses in ascending order. Note that the optimality gap is undefined if none of the nonlinear solvers succeed. The optimality gap is close to zero in many cases and below 1% in all but a handful of cases.

3.6 Scalability

We end this section by comparing the time required by each solver to solve the test cases. Fig. 2 shows the time used by the SDP solvers compared to the number of buses in the case. To make a fair comparison, we report computation times without preprocessing, *i.e.*, only the time required by the actual solver is recorded (we briefly discuss some considerations related to preprocessing in Section 4).

MOSEK is generally the fastest. The difference between MOSEK and SDPOPF (which also uses MOSEK, but based on the problem formulation compiled by YALMIP) highlights that the formulation of the SDR may have a significant impact on the computation time as well as robustness. The striking difference between MOSEK and CDCS, both in terms of computation time and accuracy, makes it hard to justify the use of first-order methods for highly sparse problems like these.

In addition to cost function value and optimality gap, tables 2–5 list the computation times (excluding preprocessing) for MOSEK and the three solvers IPOPT, KNITRO, and MIPS. MOSEK solves the SDR of all but one case with less than 25,000 buses in less than 10 minutes; the only exception is the case 4661_sdet from PGLIB (all three operating conditions). Solving this problem takes MOSEK around 80 minutes. The longer computation time required to solve this case compared to other cases with a similar number of buses can in part be explain by looking at the chordal embedding of the network graph. The largest clique is of size 242 which is similar to the case with 82,000 buses (238) and around three times the size of all other cases with less than 25,000 buses. The case with 25,000 power buses is solved in approximately an hour

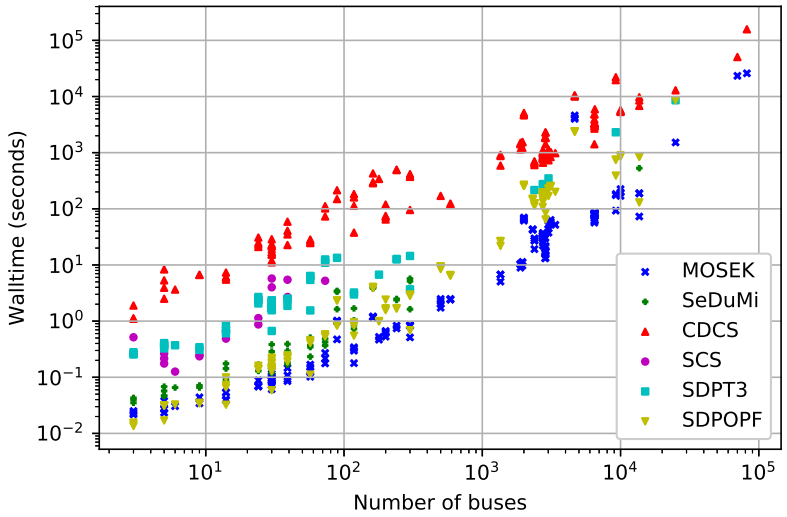


Fig. 2 Scatter plot of the time used by the SDP solvers against the number of buses for successful cases. Note that SDPOPF solves an equivalent but different SDR.

by MOSEK, and the largest cases with 70,000 and 82,000 buses are solved in around seven hours. The nonlinear solvers are typically 5-20 times faster than MOSEK (they solve a different problem!), but they sometimes fail. The RTE cases from PGLIB appear to be particularly difficult for the nonlinear solvers: in some cases, none of the nonlinear solvers succeed, and the computation times are occasionally large compared to the general trend.

4 Discussion

The difference between our formulation of the SDR and the one constructed by SDPOPF via YALMIP shows that the problem formulation can have a significant impact on computation times and robustness. Our experiments demonstrate that an SDR of the ACOPF problem can be solved accurately and reliably with the right combination of problem formulation and solver. However, it is possible that the problem formulation can be further improved. For example, as mentioned in section 2.4, the conditioning of the problem may improve with some scaling of the constraints, and this, in turn, may reduce the number of iterations and/or the computation time. We have conducted some experiments in this direction, and our preliminary results show that using MOSEK, the solution time can roughly be cut in half; the geometric mean of the speed-up obtained by means of scaling was 1.9. Indeed, the solution time for the largest test case with 25,000 buses was reduced from about one hour to half an hour with MOSEK. We did not observe a similar improvement with scaling for the other solvers. We note that SOCR and QCR implementations

could possibly benefit from scaling in a similar way. Finally, we remark that scaling may affect stopping criteria, so care must be taken when comparing the accuracy of solutions obtained with and without scaling. The QCR, proposed by Coffrin et al. (2016), provides a promising alternative to the SDR in that it is computationally cheaper and often as tight as the SDR (and in some cases even tighter). However, the findings reported in (Coffrin et al., 2016) only include SDRs of cases with less than 3,000 buses, and it is therefore unclear how the QCR and the SDR compare with respect to optimality gap for larger test cases. Moreover, the results pertaining to the SDR were obtained using an implementation based on SDPT3 and the modeling tool CVX, so the sizable gap between the two relaxations in terms of computational time will likely shrink if MOSEK and our problem formulation is used for the SDR.

The computation times reported in Section 3 did not include preprocessing time (*i.e.*, the time required to construct the SDR). To give the reader an idea of the preprocessing workload, we remark that the construction of the SDR of the case with 25 thousand buses took approximately 25 seconds or approximately 1/60 of the time required to solve the SDR with MOSEK, and the geometric average of the ratio of the solution time to the preprocessing time for cases with more than 300 buses was approximately 13, *i.e.*, preprocessing accounted for around 7% of the total time on average. In contrast, YALMIP (via SDPOPF) required approximately 6 minutes to compile the case with 25,000 buses. Comparing the ratio of the preprocessing time for YALMIP to that of our approach, we found that the geometric average was approximately 13, *i.e.*, on average it took 13 times longer with YALMIP. We note that our Python-based preprocessing code may be improved, *e.g.*, by reimplementing critical parts of the code in C. In principle, the preprocessing time may be amortized if several problem instances with the same underlying power network need to be solved. However, this would require a symbolic chordal conversion of the problem such that the problem data can easily be updated or replaced.

5 Conclusion

SDR is a promising technique that may be used to compute useful global lower bounds on the optimal value of ACOPF problems. However, concerns about robustness and scalability have cast doubt on the practical usefulness of the technique. We have shown experimentally that the problem formulation can have a significant impact on both robustness and scalability. By constructing the SDR manually instead of using a modeling tool, we avoid problem transformations that incur significant overhead. Our numerical experiments establish that SDRs of a large collection of test cases can be solved reliably with MOSEK. Moreover, the time required to solve an SDR is typically within an order of magnitude of the time required by state-of-the-art nonlinear solvers such as KNITRO and IPOPT.

References

Andersen, M. S. (2018). OPFSDR v0.2.3. https://git.io/opfsdr. Accessed on September 4 2018.

Andersen, M. S., Hansson, A., and Vandenberghe, L. (2014). Reduced-complexity semidefinite relaxations of optimal power flow problems. *IEEE Trans. Power Syst.*, 29(4):1855–1863.

Bai, X., Wei, H., Fujisawa, K., and Wang, Y. (2008). Semidefinite programming for optimal power flow problems. *International Journal of Electrical Power and Energy Systems*, 30(6-7):383–392.

Bingane, C., Anjos, M. F., and Digabel, S. L. (2018). Tight-and-cheap conic relaxation for the AC optimal power flow problem. *IEEE Trans. Power Syst.*

Birchfield, A. B., Xu, T., Gegner, K. M., Shetye, K. S., and Overbye, T. J. (2017). Grid structural characteristics as validation criteria for synthetic networks. *IEEE Transactions on Power Systems*, 32(4):3258–3265.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

Byrd, R. H., Nocedal, J., and Waltz, R. A. (2006). Knitro: An integrated package for nonlinear optimization. In *Large Scale Nonlinear Optimization, 35–59, 2006*, pages 35–59. Springer Verlag.

Carpentier, J. (1962). Contribution á l'étude du dispatching économique. *Bulletin de la Société Française des Électriciens*, 3:431–447.

Coffrin, C., Hijazi, H., and Van Hentenryck, P. (2016). The QC relaxation: A theoretical and computational study on optimal power flow. *IEEE Trans. Power Syst.*, 31(4):3008–3018.

Fukuda, M., Kojima, M., Murota, K., and Nakata, K. (2001). Exploiting sparsity in semidefinite programming via matrix completion I: General framework. *SIAM Journal on Optimization*, 11(3):647–674.

Grant, M. and Boyd, S. (2008). Graph implementations for nonsmooth convex programs. In Blondel, V., Boyd, S., and Kimura, H., editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited.

Hijazi, H., Coffrin, C., and Hentenryck, P. V. (2017). Convex quadratic relaxations for mixed-integer nonlinear programs in power systems. *Mathematical Programming Computation*, 9(3):321–367.

Hijazi, H., Coffrin, C., and Van Hentenryck, P. (2016). Polynomial SDP cuts for optimal power flow. In *19th Power Systems Computation Conference, PSCC 2016*.

Jabr, R. A. (2006). Radial distribution load flow using conic programming. *IEEE Trans. Power Syst.*, 21(3):1458–1459.

Jabr, R. A. (2012). Exploiting sparsity in SDP relaxations of the OPF problem. *IEEE Trans. Power Syst.*, 27(2):1138–1139.

Josz, C., Fliscounakis, S., Maeght, J., and Panciatici, R. (2016). AC power flow data in MATPOWER and QCQP format: iTesla, RTE Snapshots, and PEGASE. arXiv:1603.01533v3.

Josz, C., Maeght, J., Panciatici, P., and Gilbert, J. C. (2015). Application of the moment-SOS approach to global optimization of the OPF problem. *IEEE Trans. Power Syst.*, 30(1):463–470.

Kocuk, B., Dey, S. S., and Sun, X. A. (2016). Strong socp relaxations for the optimal power flow problem. *Operations Research*, 64(6):1177–1196.

Kourounis, D., Fuchs, A., and Schenk, O. (2018). Toward the next generation of multiperiod optimal power flow solvers. *IEEE Transactions on Power Systems*, 33(4):4005–4014.

Lavaei, J. and Low, S. H. (2012). Zero duality gap in optimal power flow problem. *IEEE Trans. Power Syst.*, 27(1):92–107.

Löfberg, J. (2004). YALMIP : A toolbox for modeling and optimization in MATLAB. In *In Proceedings of the CACSD Conference*, Taipei, Taiwan.

Low, S. H. (2014a). Convex relaxation of optimal power flow—part I: Formulations and equivalence. *IEEE Transactions on Control of Network Systems*, 1(1):15–27.

Low, S. H. (2014b). Convex relaxation of optimal power flow—part II: Exactness. *IEEE Transactions on Control of Network Systems*, 1(2):177–189.

Madani, R., Kalbat, A., and Lavaei, J. (2017). A low-complexity parallelizable numerical algorithm for sparse semidefinite programming.

Mak, T. W. K., Shi, L., and Hentenryck, P. V. (2018). Phase transitions for optimality gaps in optimal power flows a study on the French transmission network. arXiv:1807.05460.

Mittelmann, H. D. (2003). An independent benchmarking of SDP and SOCP solvers. *Mathematical Programming*, 95(2):407–430.

Molzahn, D. K. and Hiskens, I. A. (2015). Sparsity-exploiting moment-based relaxations of the optimal power flow problem. *IEEE Trans. Power Syst.*, 30(6):3168–3180.

Molzahn, D. K., Holzer, J. T., Lesieutre, B. C., and DeMarco, C. L. (2013). Implementation of a large-scale optimal power flow solver based on semidefinite programming. *IEEE Trans. Power Syst.*, 28(4):3987–3998.

MOSEK (2015). *MOSEK Optimizer API for Python.*

O'Donoghue, B., Chu, E., Parikh, N., and Boyd, S. (2016). Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, 169(3):1042–1068.

PGLib-OPF (2018). Power Grid Lib - Optimal Power Flow v18.08. https://git.io/pglib-opf. Accessed on September 4 2018.

Sturm, J. F. (1999). Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11(1):625–653.

Taylor, J. A. (2015). *Convex Optimization of Power Systems.* Cambridge University Press.

Toh, K. C., Todd, M. J., and Tütüncü, R. H. (1999). SDPT3 — a Matlab software package for semidefinite programming, version 1.3. *Optimization Methods and Software*, 11(1-4):545–581.

Wächter, A. and Biegler, L. T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57.

Wang, H., Murillo-Sanchez, C. E., Zimmerman, R. D., and Thomas, R. J. (2007). On computational issues of market-based optimal power flow. *IEEE Trans. Power Syst.*, 22(3):1185–1193.

Zheng, Y., Fantuzzi, G., Papachristodoulou, A., Goulart, P., and Wynn, A. (2016). CDCS: Cone Decomposition Conic Solver, version 1.1.

Zimmerman, R. D., Murillo-Sánchez, C. E., and Thomas, R. J. (2011). MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education. *IEEE Trans. Power Syst.*, 26(1):12–19.

Table 2 Cost, gap, and computation time for MATPOWER cases with more than 300 buses and without dispatchable loads. Failures are reported as 'M' (max. iterations), 'I' (termination at infeasible point), 'N' (numerical error in solver). Times shown in red correspond to failures.

		Cost				Gap	Time (sec.)			
	Case	IPOPT	KNITRO	MIPS	MOSEK		IPOPT	KNITRO	MIPS	MOSEK
MATPOWER	ACTIVSg500	7.258e+04	7.258e+04	7.258e+04	7.105e+04	2.1%	1.6	0.7	0.6	2.0
	1354pegase	7.407e+04	7.407e+04	7.407e+04	7.406e+04	0.0%	5.3	1.7	2.3	5.0
	1888rte	F	5.980e+04	F	5.960e+04	0.3%	203	17.7	1.0	8.9
	1951rte	8.174e+04	8.174e+04	F	8.173e+04	0.0%	19.9	9.7	1.2	9.7
	ACTIVSg2000	1.228e+06	1.229e+06	1.228e+06	1.228e+06	0.0%	16.2	2.5	3.3	69.9
	2383wp	1.869e+06	1.869e+06	1.869e+06	1.861e+06	0.4%	11.1	3.2	3.2	27.0
	2736sp	1.308e+06	1.308e+06	1.308e+06	1.308e+06	0.0%	27.7	2.7	3.0	29.2
	2737sop	7.776e+05	7.776e+05	7.776e+05	7.775e+05	0.0%	11.4	2.7	2.8	32.6
	2746wop	1.208e+06	1.208e+06	1.208e+06	1.208e+06	0.0%	11.5	2.8	3.2	33.7
	2746wp	1.632e+06	1.632e+06	1.632e+06	1.632e+06	0.0%	14.8	2.8	3.1	32.3
	2848rte	F	5.302e+04	F	5.301e+04	0.0%	485	31.8	1.9	13.1
	2868rte	7.979e+04	7.979e+04	F	7.979e+04	0.0%	80.7	10.7	2.4	13.3
	2869pegase	1.340e+05	1.340e+05	1.340e+05	1.340e+05	0.0%	10.7	2.9	4.6	15.2
	3012wp	2.592e+06	2.592e+06	2.592e+06	2.588e+06	0.1%	48.5	3.3	5.0	44.2
	3120sp	2.143e+06	2.143e+06	2.143e+06	2.142e+06	0.0%	16.3	3.2	5.3	53.9
	3375wp	7.412e+06	7.412e+06	7.412e+06	7.409e+06	0.0%	15.8	3.6	5.9	51.9
	6468rte	8.683e+04	8.683e+04	F	8.682e+04	0.0%	56.3	23.5	10.7	56.7
	6470rte	9.835e+04	9.835e+04	F	9.834e+04	0.0%	297	26.9	11.7	60.4
	6495rte	1.063e+05	1.063e+05	F	1.061e+05	0.2%	198	25.0	8.7	61.7
	6515rte	1.098e+05	1.098e+05	F	1.097e+05	0.1%	279	46.6	10.4	62.8
	9241pegase	3.159e+05	3.159e+05	3.159e+05	3.158e+05	0.0%	208	131	19.8	92.8
	ACTIVSg10k	2.486e+06	2.486e+06	F	2.486e+06	-0.0%	73.2	151	42.5	170
	13659pegase	F	3.861e+05	F	3.861e+05	0.0%	1,221	72.3	3,953	72.9
	ACTIVSg25k	6.018e+06	6.018e+06	F	6.017e+06	0.0%	405	51.2	85.9	1,520
	ACTIVSg70k	1.644e+07	1.644e+07	F	1.644e+07	0.0%	896	199	140	23,343
	SyntheticUSA	F	F	F	2.017e+07	—	8,582	10,534	439	25,922

Table 3 Cost, gap, and computation time for PGLIB cases in typical operating condition with more than 300 buses and without dispatchable loads. Failures are reported as 'M' (max. iterations), 'I' (termination at infeasible point), 'N' (numerical error in solver). Times shown in red correspond to failures.

		Cost				Gap	Time (sec.)			
	Case	IPOPT	KNITRO	MIPS	MOSEK		IPOPT	KNITRO	MIPS	MOSEK
PGLIB	500_tamu	7.258e+04	7.258e+04	7.258e+04	7.105e+04	2.1%	1.8	0.6	0.5	2.3
	588_sdet	3.816e+05	3.816e+05	3.816e+05	3.798e+05	0.4%	1.4	0.8	1.1	2.4
	1354_pegase	1.364e+06	1.364e+06	1.364e+06	1.356e+06	0.6%	5.0	2.1	2.7	6.7
	1888_rte	1.640e+06	1.565e+06	F	1.538e+06	1.7%	52.0	35.6	3.9	10.2
	1951_rte	2.375e+06	F	F	2.375e+06	0.0%	39.6	113	1.0	11.0
	2000_tamu	1.228e+06	1.228e+06	1.228e+06	1.228e+06	0.0%	17.0	2.6	3.7	65.2
	2316_sdet	2.257e+06	2.257e+06	2.257e+06	2.240e+06	0.7%	8.5	3.1	4.5	43.1
	2383wp_k	1.869e+06	1.869e+06	1.869e+06	1.861e+06	0.4%	11.8	4.0	3.2	27.4
	2736sp_k	1.308e+06	1.308e+06	1.308e+06	1.308e+06	0.0%	12.8	2.8	3.2	29.8
	2737sop_k	7.776e+05	7.776e+05	7.776e+05	7.775e+05	0.0%	10.9	3.0	3.0	31.1
	2746wp_k	1.632e+06	1.632e+06	1.632e+06	1.632e+06	0.0%	14.4	3.5	3.3	34.9
	2746wop_k	1.208e+06	1.208e+06	1.208e+06	1.208e+06	0.0%	12.5	3.6	3.6	36.7
	2848_rte	1.385e+06	1.385e+06	F	1.384e+06	0.0%	75.4	12.4	13.9	16.4
	2853_sdet	F	2.469e+06	2.469e+06	2.456e+06	0.5%	46.9	4.7	6.8	29.8
	2868_rte	2.260e+06	2.260e+06	F	2.260e+06	-0.0%	43.1	18.7	18.0	17.4
	2869_pegase	2.605e+06	2.605e+06	2.605e+06	2.603e+06	0.1%	20.8	5.2	6.9	20.7
	3012wp_k	2.601e+06	2.601e+06	2.601e+06	2.597e+06	0.1%	21.0	4.0	5.4	50.7
	3120sp_k	2.146e+06	2.146e+06	2.146e+06	2.145e+06	0.0%	17.3	4.0	5.7	58.4
	4661_sdet	F	2.786e+06	F	2.768e+06	0.6%	554	8.9	19.7	4,184
	6468_rte	F	2.262e+06	F	2.252e+06	0.5%	1,476	76.5	3.5	82.5
	6470_rte	F	F	F	2.545e+06	—	1,434	31.5	7.6	77.7
	6495_rte	3.478e+06	F	F	2.966e+06	14.7%	489	153	44.3	77.1
	6515_rte	F	3.197e+06	F	2.992e+06	6.4%	1,403	61.4	39.3	84.0
	9241_pegase	6.775e+06	6.775e+06	F	6.770e+06	0.1%	399	27.9	75.4	175
	10000_tamu	2.486e+06	2.486e+06	F	2.486e+06	-0.0%	98.7	113	40.1	195
	13659_pegase	1.078e+07	1.078e+07	1.078e+07	1.078e+07	0.0%	250	66.5	54.6	190

Table 4 Cost, gap, and computation time for PGLIB cases with small angle differences with more than 300 buses and without dispatchable loads. Failures are reported as 'M' (max. iterations), 'I' (termination at infeasible point), 'N' (numerical error in solver). Times shown in red correspond to failures.

		Cost				Gap	Time (sec.)			
	Case	IPOPT	KNITRO	MIPS	MOSEK		IPOPT	KNITRO	MIPS	MOSEK
PGLIB SAD	500_tamu	7.923e+04	7.923e+04	7.923e+04	7.322e+04	7.6%	2.7	0.7	0.6	2.5
	588_sdet	4.043e+05	4.043e+05	4.043e+05	3.814e+05	5.6%	2.1	0.8	1.2	2.5
	1354_pegase	1.365e+06	1.365e+06	1.365e+06	1.357e+06	0.6%	5.8	2.3	2.7	6.7
	1888_rte	F	1.640e+06	F	1.538e+06	6.2%	331	17.3	3.4	10.4
	1951_rte	2.383e+06	F	F	2.376e+06	0.3%	54.5	43.9	1.1	11.3
	2000_tamu	1.230e+06	1.230e+06	1.230e+06	1.229e+06	0.1%	31.2	3.2	3.9	61.4
	2316_sdet	2.257e+06	2.257e+06	2.257e+06	2.240e+06	0.7%	9.2	3.8	4.5	42.1
	2383wp_k	1.916e+06	1.916e+06	1.916e+06	1.905e+06	0.6%	15.8	3.9	3.4	29.8
	2736sp_k	1.329e+06	1.329e+06	1.329e+06	1.325e+06	0.4%	15.7	3.8	4.0	33.2
	2737sop_k	7.927e+05	7.927e+05	7.927e+05	7.859e+05	0.9%	15.8	4.2	3.8	33.4
	2746wp_k	1.667e+06	1.667e+06	1.667e+06	1.661e+06	0.4%	15.8	4.4	3.7	35.8
	2746wop_k	1.234e+06	1.234e+06	1.234e+06	1.226e+06	0.7%	16.7	3.9	3.8	36.9
	2848_rte	F	F	F	1.385e+06	—	558	106	13.0	16.4
	2853_sdet	F	2.495e+06	2.495e+06	2.458e+06	1.5%	87.5	5.4	6.5	30.7
	2868_rte	F	F	F	2.264e+06	—	41.3	21.7	2.7	17.4
	2869_pegase	2.620e+06	2.620e+06	2.620e+06	2.616e+06	0.2%	22.3	6.2	8.1	21.8
	3012wp_k	2.621e+06	2.621e+06	2.621e+06	2.610e+06	0.4%	21.9	5.2	6.0	52.7
	3120sp_k	2.176e+06	2.176e+06	2.176e+06	2.165e+06	0.5%	21.8	5.8	6.0	63.4
	4661_sdet	F	2.802e+06	2.802e+06	2.781e+06	0.7%	459	8.8	19.4	4,005
	6468_rte	2.262e+06	2.262e+06	F	2.252e+06	0.5%	434	45.2	3.5	80.9
	6470_rte	F	F	F	2.548e+06	—	2,637	1,519	16.1	78.6
	6495_rte	F	3.478e+06	F	2.966e+06	14.7%	2,458	54.2	22.7	82.0
	6515_rte	F	F	F	2.992e+06	—	1,234	143	6.1	79.3
	9241_pegase	6.920e+06	6.920e+06	F	6.823e+06	1.4%	142	29.1	89.5	183
	10000_tamu	F	2.486e+06	F	2.486e+06	-0.0%	1,370	94.6	38.9	196
	13659_pegase	1.090e+07	1.090e+07	1.090e+07	1.082e+07	0.7%	187	42.7	55.5	188

Table 5 Cost, gap, and computation time for heavily loaded PGLIB cases (i.e., binding thermal limits) with more than 300 buses and without dispatchable loads. Failures are reported as 'M' (max. iterations), 'I' (termination at infeasible point), 'N' (numerical error in solver). Times shown in red correspond to failures.

		Cost				Gap	Time (sec.)			
	Case	IPOPT	KNITRO	MIPS	MOSEK		IPOPT	KNITRO	MIPS	MOSEK
PGLIB API	500_tamu	4.034e+04	4.034e+04	4.034e+04	4.034e+04	-0.0%	1.1	0.6	0.5	1.7
	588_sdet	4.996e+05	4.996e+05	F	4.983e+05	0.3%	1.4	0.8	0.4	2.4
	1888_rte	F	2.262e+06	F	2.259e+06	0.2%	672	8.0	4.1	10.6
	2000_tamu	1.288e+06	1.288e+06	1.288e+06	1.275e+06	1.0%	25.7	11.8	4.2	67.6
	2316_sdet	2.774e+06	2.774e+06	2.774e+06	2.758e+06	0.6%	9.1	3.8	4.1	43.2
	2383wp_k	2.791e+05	2.791e+05	2.791e+05	2.791e+05	0.0%	7.0	1.9	1.7	19.2
	2736sp_k	6.260e+05	6.260e+05	6.260e+05	6.097e+05	2.6%	14.7	4.3	3.5	30.7
	2737sop_k	3.587e+05	3.587e+05	3.587e+05	3.485e+05	2.8%	12.9	4.0	3.1	30.6
	2746wp_k	5.818e+05	5.818e+05	5.818e+05	5.818e+05	0.0%	7.6	2.6	2.2	22.1
	2746wop_k	5.117e+05	5.117e+05	5.117e+05	5.117e+05	0.0%	6.9	2.3	1.7	24.8
	3012wp_k	7.289e+05	7.289e+05	7.289e+05	7.289e+05	0.0%	10.8	2.6	5.2	37.5
	3120sp_k	F	9.696e+05	9.696e+05	8.818e+05	9.1%	43.8	5.3	5.8	57.5
	4661_sdet	F	3.343e+06	F	3.319e+06	0.7%	207	9.7	5.7	4,604
	6468_rte	F	F	F	2.718e+06	—	1,543	172	13.2	84.2
	6470_rte	F	F	F	3.174e+06	—	1,319	859	10.0	76.1
	6495_rte	F	F	F	3.735e+06	—	2,045	111	8.5	82.1
	6515_rte	F	F	F	3.657e+06	—	2,650	54.4	5.7	85.6
	10000_tamu	1.816e+06	1.816e+06	F	1.751e+06	3.6%	153	94.0	20.1	225